CSCI 136
Data Structures & Advanced Programming

Morgan McGuire
Bill Jannen
Jon Park
Feb 3, 2017
Today’s Outline

• Course Overview
• Java refresher
Administrative Details

• Sign in today!
• Wait list & survey
• 1 Handout: Syllabus

• Instructors:
  • **Morgan**: TCL 308, mcguire@cs.williams.edu
  • **Bill**: TCL 307, jannen@cs.williams.edu
  • **Jon**: TCL 209, jpark@cs.williams.edu
Course Information

• **Lecture**
  • MWF 9-9:50am or 10-10:50am
  • TCL 202

• **Lab**
  • Wed 1-4pm, Wed 7-10pm, or Thu 1-4pm
  • TCL 217a or TCL 216
  • Door code: [redacted]

• **Webpage:** [https://www.cs.williams.edu/~cs136](https://www.cs.williams.edu/~cs136)

• **Textbook**
  • Java Structures: Data Structures in Java for the Principled Programmer, √7 Edition (by Duane Bailey)
  • Optional hardcopy from Lauren Vining in TCL 303
CS136 Scope

• Data Structures
  • Common ways to store and manipulate data

• Advanced Programming
  • Use structures to write programs that solve (interesting) problems
CS136 Goals

• Identify basic data structures
  • Examples?
  • list, stack, array, tree
• Implement these structures in Java
• Learn how to evaluate and visualize data structures
  • Linked lists and vectors both represent lists of items
  • Different representations of data
  • Different algorithms for manipulating/accessing/storing data
• Learn how to design large programs that are easy to modify, extend, and debug
• Have fun!
Example Programs

• Find a way to drive from Williamstown, MA to San Diego, CA
• Find the **shortest** way to drive from Williamstown, MA to San Diego, CA
• Schedule a flight with the fewest layovers from Albany, NY to Beijing, China
• Schedule exams so no students have conflicts
Common Themes

1. Identify data for problem
2. Identify questions to answer about data
3. Design data structures and algorithms to solve questions correctly and efficiently (Note: not all correct solutions are efficient, and vice versa!)
4. Implement solutions that are robust, adaptable, and reusable

Example: Boggle
Course Outline

1. **Java crash course**
2. **Foundations of programming**
   - Vocabulary
   - Analysis tools
   - Recursion
   - Methodology
3. **Basic structures**
   - Lists, vectors, queues, stacks
4. **Advanced structures**
   - Graphs, heaps, trees, dictionaries
Why CS136 == Awesome

- CS134/135 teach the basics of programming
- CS136 unlocks the power of computation to solve problems in CS, econ, math, biology, physics, etc.
  - Data representation
  - Algorithms
  - Abstraction
  - Software design
  - Elegance
  - **Scalability**: memory, performance, people
Course Policies
Honor Code and Ethics

• The student handbook describes the Honor Code and Computer Ethics guidelines.

• You should also know the CS Dept computer usage policy.
  • If you are not familiar with these items, please review them.

• We take these things very seriously…
Your Responsibilities

- Come to lab and lecture on time.
- Read assigned chapters before class and lab.
  - Bring textbook to lab (or be prepared to use PDF)
- **Come to lab prepared!!!**
  - Bring design docs for program
  - 1 prof + several TAs = lots of attention for you—take advantage of this!
- Do NOT remain confused. Get help.
- Don’t cheat.
- Participate in class discussions.
Programming Review & Java Crash Course, Part I
Why Java?

• There are lots of programming languages…
  • C, Pascal, C++, Java, C#, Python
• Java was designed in 1990s to ease Internet programming
• Java is good because:
  • It’s easy (well, easier than predecessors like C++) to write correct programs
  • Object-oriented – good for large systems
  • Easy support for abstraction
This semester, we will...

- ...NOT use BlueJ or Python. :-)  
- ...focus more on structures and algorithms than on graphics, networks, etc. (we can always add graphics later)  
- ...use Emacs and Java.  
- ...compile from a terminal rather than by clicking a button.
Java Crash Course Goals

• Review the big ideas
• Use lab/book/TAs/web to fill in any gaps
• Don’t hesitate to ask me or the TAs for a refresher on any topic
• We’re going to go fast…interrupt anytime
Java

• **Variable types**
  • int, double, boolean, String, …

• **Statements**
  • int x = 3;
  • x = x + 2;
  • if (x > 3) { … } else { … }
  • while (x < 2) { … }
  • for (int i = 0; i < x; i++) { …}

• **Comments**
  • //this is a comment
  • /* so is this */
Sample Programs

- **Hello.java**
  - Write a program that prints “Hello” to the terminal.
Sample Programs

- Hello.java
  - Write a program that prints “Hello” to the terminal.

- Sum.java
  - Write a program that adds two integers together and returns the sum
    - Command-line args
    - Using Scanner
Next time...

• We’ll continue reviewing Java
  • Object-oriented programming